Continuous and Embedded Learning in Autonomous Vehicles: Adapting to Sensor Failures

Alan C. Schultz[a], and John J. Grefenstette[b]

[a]Intelligent Systems Section, Naval Research Laboratory, Washington DC, USA

[b]Institute for Biosciences, Bioinformatics and Biotechnology, George Mason University, Fairfax VA, USA

ABSTRACT

This project describes an approach to creating autonomous systems that can continue to learn throughout their lives, that is, to be adaptive to changes in the environment and in their own capabilities. Evolutionary learning methods have been found to be useful in several areas in the development of autonomous vehicles. In our research, evolutionary algorithms are used to explore alternative robot behaviors within a simulation model as a way of reducing the overall knowledge engineering effort. The learned behaviors are then tested in the actual robot and the results compared. Initial research demonstrated the ability to learn reasonable complex robot behaviors such as herding, and navigation and collision avoidance using this offline learning approach. In this work, the vehicle is always exploring different strategies via an internal simulation model; the simulation, in turn, is changing over time to better match the world.

This model, which we call *Continuous and Embedded Learning* (also referred to as *Anytime Learning*), is a general approach to continuous learning in a changing environment. The agent's learning module continuously tests new strategies against a simulation model of the task environment, and dynamically updates the knowledge base used by the agent on the basis of the results. The execution module controls the agent's interaction with the environment, and includes a monitor that can dynamically modify the simulation model based on its observations of the environment. When the simulation model is modified, the learning process continues on the modified model. The learning system is assumed to operate indefinitely, and the execution system uses the results of learning as they become available. Early experimental studies demonstrate a robot that can learn to adapt to failures in its sonar sensors.

Keywords: Robotics, Learning, Genetic Algorithms, adaptation, Evolutionary Algorithms

1. INTRODUCTION

An important problem arising in robots that are expected to perform autonomously for extended periods is how to adapt the robot's rules of behavior in response to unexpected changes in its own capabilities. For example, suppose the robot periodically checks its sensors and its ability to perform basic actions. If it finds that some sensors or actions are no longer available, perhaps due to a problem with the robot's hardware or due to some undetected environmental cause, then it must learn new rules for performing its mission that use whatever remaining capabilities are still available.

We have developed an approach to this problem that we call *Anytime Learning*,[1–3] or more recently, *Continuous and Embedded Learning (CEL)*. In this approach, the robot interacts both with the external environment and with an internal simulation. The robot's execution module controls the robot's interaction with the environment, and includes a monitor that dynamically modifies the robot's internal simulation model based on the monitor's observations of the actual robot and the sensed environment. The robot's learning module continuously tests new strategies for the robot against the simulation model, using a genetic algorithm[4] to evolve improved strategies, and updates the knowledge base used by the execution module with the best available results. Whenever the simulation model is modified due to some observed change in the robot or the environment, the genetic algorithm is restarted on the modified model. The learning system operates indefinitely, and the execution system uses the results of learning as they become available.

Send correspondence to schultz@aic.nrl.navy.mil. Appeared in Unmanned Ground Vehicle Technology II, (Eds. Grant R. Gerhart, Robert W. Gunderson, Chuck M. Shoemaker), Proceedings of SPIE Vol.4024, pg. 55-62, 2000.

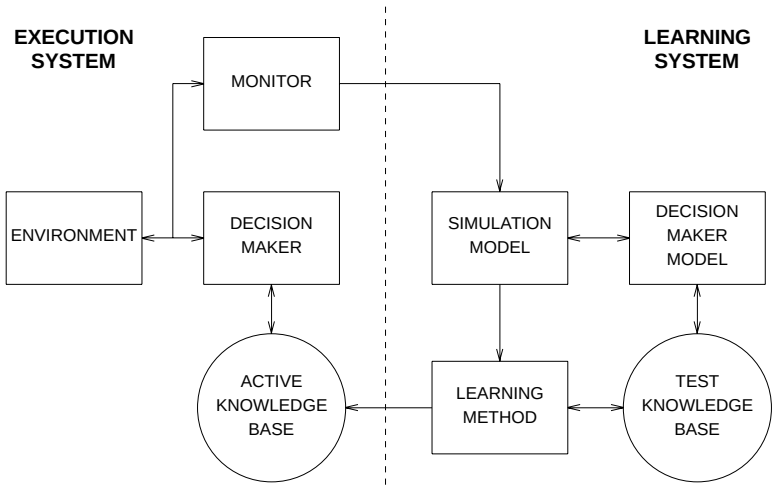

Figure 1. The Continuous and Embedded Learning Model

In this paper, we examine the CEL model with respect to sensor failures, specifically, how the robot can learn to adapt to failures in its sensor capabilities over time. We show that a robot adapt to the partial loss of its sensors and learn to use different sensors to continue to perform a door traversal task. A simulation study and execution on an actual mobile robot shows that the approach yields effective adaptation to a variety of partial sensor failures.

In the next section, we will describe the Continuous and Embedded Learning model. Section 3 discusses related work. In Section 4 we will describe the task domain. This is followed by description of the modules of the CEL model, and simulation and experimental results showing the adaptation of a robot to partial sensor failues.

2. CONTINUOUS AND EMBEDDED LEARNING

The Continuous and Embedded Learning model addresses the problem of adapting a robot's behavior in response to changes in its operating environment and its capabilities. The outline of the approach is shown in Figure 1.

There are two main modules in the CEL model. The *execution module* controls the robot's interaction with its environment. The *learning module* continuously tests new strategies for the robot against a simulation model of the environment. When the learning module discovers a new strategy that, based on simulation runs, appears to be likely to improve the robot's performance, it updates the rules used by the execution module. The execution module includes a *monitor* that measures aspects of the operational environment and the robot's own capabilities, and dynamically modifies the robot's internal simulation model based on these observations. When the monitor modifies the simulation because of an environmental change, it notifies the learning system to restart its learning process on the new simulation.

This general architecture may be implemented using a wide variety of execution modules, learning methods, and monitors. The key characteristics of the approach are:

- Learning continues indefinitely. This is unlike most machine learning methods, which employ a training phase, followed by a performance phase in which learning is disabled. This lifetime learning is what allows the system to be adaptive after being fielded.
- The learning system experiments on a simulation model. For most real-world robotic applications, experimenting with the physical robot may be time-consuming or dangerous. Using a simulation models permits the safe use of learning methods that consider strategies that may occasionally fail.
- The simulation model is updated to reflect changes in the real robot or environment. This is a secondary type of learning of the model. While this is a major research issue in its own right, we are currently not concerned with the learning at this level, and construct the monitor and simulation as appropriate. That is, we assume that it is possible to monitor the condition of the robot's sensors and actuators. For our purposes, it is not

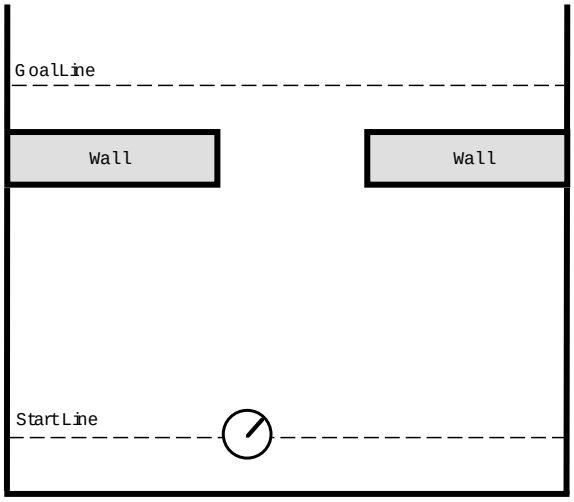

Figure 2. The door traversal task.

necessary to diagnose the cause of any detected failure, only the symptoms. We assume that the simulation has been constructed to allow the instantiation of sensor and actuator failure modes.

This final point reflects our assumption that the robot designer generally has at least partial knowledge of the robot and the environment. Knowledge that is relatively certain can be embodied in the fixed part of the simulation. Such knowledge might include certain fixed characteristics of the physical environment (e.g., gravity), as well as some aspects of the robot's design and performance. On the other hand, the robot designer should also identify those aspects of the environment and the robot's capabilities that are uncertain, and include these as changeable parts of the simulation module.

3. RELATED WORK

There has been a great deal of work done in the area of evolution of behaviors for autonomous robots. A very common approach has been in the evolution of neural controllers for robots.[5,6] An alternative approach has been the evolution of stimulus-response rules.[7]

An interesting problem has been how to add on-line adaptation to these systems in order to handle problems such as changes in the environment and to the robots capabilities. Early interesting work includes systems where both the form and the function are coevolved,[8–11] but these systems have not performed studies yet on actual robots, nor are these techniques useful for on-line performance.

Another very interesting area in the use of evolutionary algorithms on-line on real robots.[12] While this does allow a robot to be adaptive during performance, it is not clear whether this is appropriate for large robots in the real world, where trial and error experimentation can lead to damage or dangerous behavior. Our approach solves this problem by only allowing experimentation on a simulation that is internal to the robot.

4. PERFORMANCE TASK

This task requres a robot to go from one side of a room to the other, passing through an opening in a wall placed across the room, as illustrated in Figure 2.

In each trial, the robot is placed randomly along the starting line four feet in front of the back wall, facing in a randomly selected direction from -90 to 90 degrees (with 0 degrees facing the goal). The center of the front wall is located 12.5 feet from the back wall. The room is 25 feet wide. The location of the six foot opening in the front wall is also randomly selected each trial.

The robot must then reactively navigate through the opening reaching the goal line one foot beyond the wall, by learning a set of rules which map the current sensors to the actions to be performed by the robot, at a one hertz decision rate. The robot has a limited time to perform the task. Exceeding the time limit, or having a collision with any of the walls ends the current trial.

The robot used in these experiments is a Nomadic Technologies Nomad 200 Mobile Robot, a three-wheeled, synchronized-steering vehicle. The internal simulation used by the robot for learning approximated the robots sensors and effectors.

5. EXECUTION MODULE

The execution module for the robot includes a rule-based system that operates on reactive (stimulus-response) rules. A typical rule might be:

IF range = [35, 45] AND *front_sonar* < 20 AND *right_sonar* > 50 THEN SET turn = -24 (Strength 0.8)

Each decision cycle, the execution system compares the left hand side of each rule to the current sensor readings, selecting the best rule (after conflict resolution). That rule's action is then executed causing the robot to move. This is repeated until the robot succeeds or fails at the task.

In this task, the robot uses its seven front most facing sonars. Each sonar has a angular resolution of 22.5 degrees, with the front most sonar facing directly ahead, giving the robot a total sonar coverage of 157.5 degrees. We designate these sonars as *far_left*, *left*, *front_left*, *front*, *front_right*, *right*, and *far_right*. Each sonar has a maximum range of approximately 14 feet. The sonar values are all discretized; they are partitioned into intervals of 24 inches. In addition to the sonars, the robot also has a sensor that gives the range to the goal from 0 to 14 feet in 5 inch intervals, and the robots heading within the room from 0 to 359 degrees in 22.5 degree intervals.

The learned actions are velocity mode commands for controlling the translational rate and the steering rate of the robot. The translation rate is given as -4 to 10 inches/sec in 2 inch/sec intervals. The steering command is given in intervals of -10 degrees/sec from -30 to 30 degrees/sec. At each decision step, the system must choose a turning rate and steering rate based on the current sensors.

The rule *strength* is set by the learning system to estimate the quality of the rule. The execution module uses rule strengths to resolve conflicts among multiple rules that match the current sensors readings, but suggest different actions. In such cases, rules with higher strength are favored. See[13] for details.

5.1. The Monitor

In this study, the monitor periodically measures the output from the sonars, and compares them to recent readings and to the direction of motion. If the robot is moving forward, and the value of the sonar reads zero repeatedly, that particular sonar is marked as being defective. The monitor then modifies the simulation used by the learning system to replicate the failed sonar.

It is important to note that the monitor is required only to identify symptoms of problems, not the causes.

6. LEARNING MODULE

The learning module uses SAMUEL,[13] a learning program that uses genetic algorithms and other competition-based heuristics to improve its decision-making rules. Each individual in SAMUEL's genetic algorithm is an entire rule set, or strategy, for the robot. We have previously reported on using SAMUEL to learn simple robot behaviors such as navigation and collision avoidance,[14,15] robot herding,[16] and in other complex domains.

When the monitor notices a failure in any of the seven sonars, the learning module's population is re-initialized and continues with the modified simulation model. In this study, we use 50% of the population at the time of the detected failure and replace the other 50% with copies of the initial rule set (generally, go toward the goal line).

We have also used a *case-based approach* to re-initializing the population. The learning system re-initializes the population of strategies in the genetic algorithm by finding nearest neighbors from the case base consisting of previously learned strategies. Strategies in the case base are indexed by the capability list in place at the time the strategy was learned. Using previously learned strategies to initialize the population allows the system to very quickly adapt to situations that are similar to those seen before. See[2] for more details.

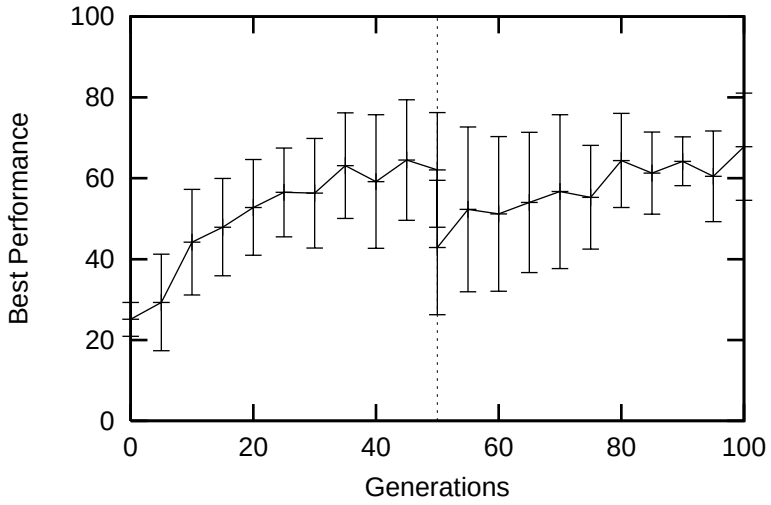

Figure 3. Learning curve showing adaptation to sensor failures occurring at generation 50. Best performance indicates the success rate for the best strategies in the current population. Results averaged over 10 runs.

7. EXPERIMENTAL METHODOLOGY

The robot begins with a set of default rules for moving toward to goal line, which give an initial success rate of 25% of getting through the doorway. These initial rules basically give the robot the direction to the goal line, but say nothing about obstacles or the walls themselves. The learning system starts with a simulation model that includeds all sonars working.

After an initial period of learning, one or more sonars are then blinded. In simulation, the failed sonar was modeled as a constant, minimum-value sonar reading; on the actual robot, one of the sonar sensors would be covered with rigid material. Once the monitor detects the failed sensor, the learning simulation is adjusted to reflect the failure, the population of competing strategies is re-initialized as described previously, and learning continues. The online robot uses the best rules discovered by the learning systems since the last change to the learning simulation model.

We initially ran an experiment with the robot adapting to a blinded *front* sonar after 50 generations, but the performance dropped a non-significant amount before continuing to improve. This is indicative of the robust rules that the SAMUEL system tends to learn. In a second experiment, we blinded the *front* sonar and the *front_right* sonar, still without a significant drop in performance. In the results reported here, three sensors were failed (*front*, *front_right*, and *right*). After the sensors were blinded, the system was allowed to continue for another 50 generations.

In these experiments we used threshold selection in the evolutionary algorithms in which the top 50% of the population reproduces (producing two offspring each). The payoff function was based on the time it took for the robot to reach the goal line. A collision resulted a a very low payoff, and exceeding the time limit resulted in a low payoff.

8. RESULTS

8.1. Quantitative Results

The experiment was repeated ten times. Figure 3 shows the average performance over time for these ten runs. The x-axis shows the generation, and the y-axis shows the external performance – the number of times out of 100 that the robot succedded in getting through the opening. The dotted vertical line at generation 50 shows where the sensor failed.

As can be seen from this learning curve, the robot initially learns to improve its performance in this task from 25% to 63% with all seven sonar sensors operating. At the beginning of generation 50, the three sonars (front, front right, and right) fail. The performance then drops to 37%, but then continues to improve as the monitor identifies the failed sonars, modifies the simulation, and learning continues with the new simulation model.

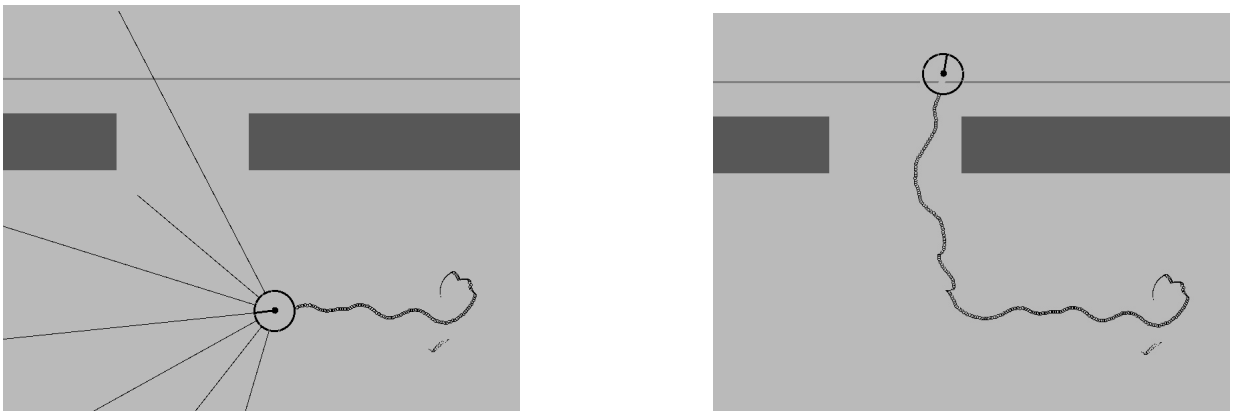

Figure 4. Robot in motion with all sensors intact, a) during run and b) at goal.

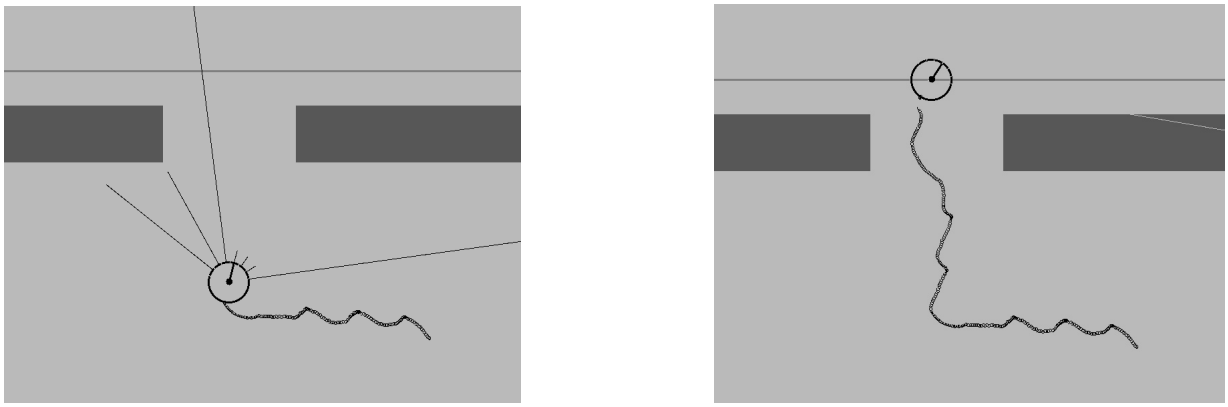

Figure 5. Robot in motion after adapting to loss of three sensors, *front*, *front_right* and *right*, a) during run, and b) at goal.

8.2. Qualitative Results

Qualitatively, we observed the following behaviors. Figure 4 shows the simulation with all of the sensors working. The picture on the left shows the simulated robot during the run, while the picture on the right shows the robot reaching the goal. In Figure 5, we see the simulated robot with three failed sonars as described above. The picture on the left shows the robot during the run, while the picture on the right shows the robot at the goal. Note that interesting behavior in the second set of pictures, as indicated by the trail showing the robot's path. The robot uses a swaying motion in order to sweep its working sensors across the space in front of the robot. This behavior allows the robot to perform the task suucessfully.

In the experimental runs we performed, when the forward facing sensors are blocked, the robot initially responds by refusing to move forward. This is fail-safe kind of response, as it allows the offline learning system to come up with a better strategy while the online robot just sits there thinking.

Figure 6 shows shots of the actual robots. The first picture shows the robot finding the opening with its front sonar, and proccedding straight at and through the opening. The second picture shows the front sonar covered to simulate its failure. The third picture shows the robot solving the task after adapting to a sonar failure. Note that it is now using a side sonar to find the opening and then turns towards the opening.

9. DISCUSSION

This work shows that the Continuous and Embedded Learning model is a promising approach to adapting to partial sensor failures. When the monitor detects a sensor failure, it modifies the system's learning simulation. The learning system operates indefinitely, and the execution system uses the results of learning as they become available. Combined

Figure 6. a) Robot with full sensors passing directly through doorway. b) Robot with front sonar covered. c) Robot after adapting to covered sonar. It uses side sonar to find opening, and then turns into the opening.

with our previous work showing adaptation to changing environments and actuator failures,[1–3] this work indicates the generality of the CEL model for the design of robust autonomous robot systems. Future work will investigate the ability to adapt to combinations of sensor and actuator failures, and attempt to quantify the limits of adaptability under this model.

ACKNOWLEDGMENTS

The work reported here was supported by The Office of Naval Research. We would like to thank Magdalena Bugajska for her assistance in preparing some of the figures.

REFERENCES

1. J. J. Grefenstette and C. L. Ramsey, "An approach to anytime learning," in *Proc. Ninth International Conference on Machine Learning*, pp. 189–195, Morgan Kaufmann, (San Mateo, CA), 1992.
2. C. L. Ramsey and J. J. Grefenstette, "Case-based anytime learning," in *Case Based Reasoning: Papers from the 1994 Workshop*, D. W. Aha, ed., Technical Report WS-94-07, AAAI Press, (Menlo Park, CA), 1994.
3. J. J. Grefenstette, "Genetic learning for adaptation in autonomous robots," in *Robotics and Manufacturing: Recent Trends in Research and Applications, Volume 6*, pp. 265–270, ASME Press, (New York), 1996.
4. J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.
5. S. Nolfi, D. Floreano, O. Mighno, and F. Mondada, "How to evolve autonomous robots: Different approaches in evolutionary robotics," in *Proc. of the International Conference on Artificial Live IV*, R. Brooks and P. Maes, eds., pp. 190–197, MIT Press, 1994.
6. I. Harvey, P. Husbands, and D. Cliff, "Issues in evolutionary robotics," in *Proceedings of the Second International Conference on Simulation of Adaptive Behaviour*, MIT Press Bradford Books, 1993.
7. A. C. Schultz, "Using a genetic algorithm to learn strategies for collision avoidance and local navigation," in *Proc. Seventh International Symposium on Unmanned Untethered Submersible Technology*, pp. 213–215, University of New Hampshire Marine Systems Engineering Laboratory, 1991.
8. K. Sims, "Evolving 3d morphology and behavior by competition," in *Proceedings of the International Conference Artificial Live IV*, pp. 28–39, MIT Press, (Cambridge, MA), 1994.
9. W.-P. Lee, J. Hallam, and H. H. Lund, "A hybrid GP/GA approach for co-evolving controllers and robot bodies to achieve fitness-specific tasks," in *Proc. of IEEE Third International Conference on Evolutionary Computation*, IEEE Press, (NJ), 1996.
10. H. H. Lund and O. Miglino, "Evolving and breeding robots," in *Proc. of the First European Workshop on Evolutionary Robotics*, Springer-Verlag, 1998.
11. M. D. Bugajska and A. C. Schultz, "Co-evolution of form and function in the design of autonomous agents: Micro air vehicle project," in *Proc. Workshop on Evolution of Sensors GECCO 2000*, IEEE, 2000.
12. D. Floreano and F. Mondada, "Evolution of homing navigation in a real mobile robot," *IEEE Transactions on System Man and Cybernetics* **26**(3), pp. 396–407, 1996.

13. J. J. Grefenstette, C. L. Ramsey, and A. C. Schultz, "Learning sequential decision rules using simulation models and competition," *Machine Learning* **5**(4), pp. 355–381, 1990.
14. A. C. Schultz and J. J. Grefenstette, "Using a genetic algorithm to learn behaviors for autonomous vehicles," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, (Hilton Head, SC), 1992.
15. A. C. Schultz, "Learning robot behaviors using genetic algorithms," in *Intelligent Automation and Soft Computing: Trends in Research, Development, and Applications*, pp. 607–612, TSI Press, (Albuquerque), 1994.
16. A. C. Schultz and J. J. Grefenstette, "Robo-shepherd: Learning complex robotic behaviors," in *Robotics and Manufacturing: Recent Trends in Research and Applications, Volume 6*, pp. 763–768, ASME Press, (New York), 1996.